Hochschule Düsseldorf
University of Applied Sciences
HSD
ZDD - DAISY

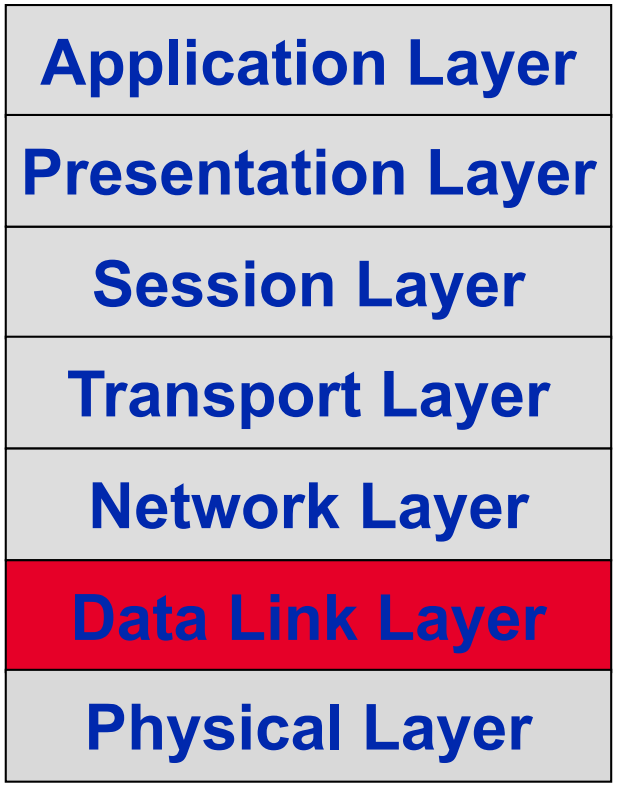
UbiComp – Teil 5: Netzwerktechnik und industrielle Kommunikation II

Prof. Dr.-Ing. Dorothea Schwung

Lernziele Teil 5

1. Sie können das Kommunikationsmodell nach Shannon erläutern.
2. Sie können mögliche Störquellen benennen.
3. Sie sind in der Lage die Bitfehlerrate zu berechnen.
4. Sie wissen, was die Hamming-Distanz ist.
5. Sie sind mit den Methoden zur Fehlererkennung vertraut.

Schicht 2 – Die Sicherungsschicht

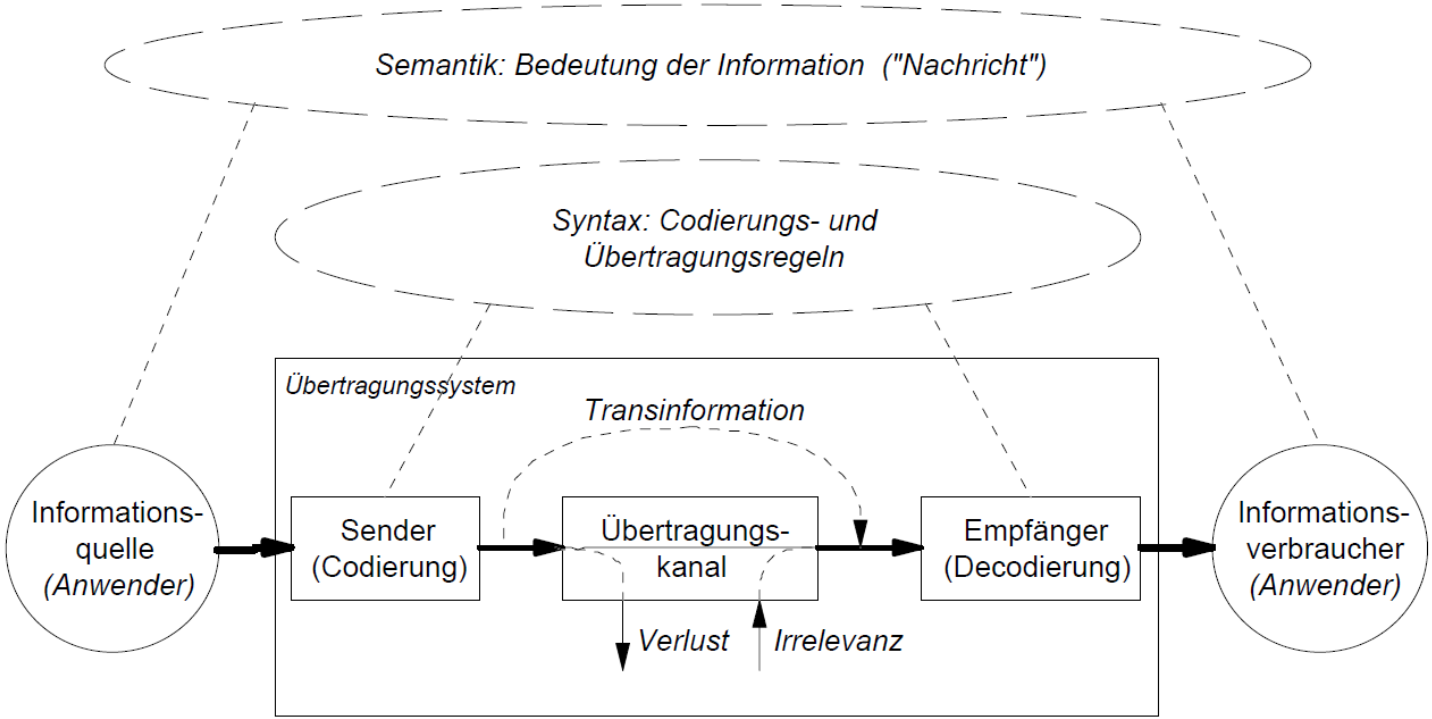


Schicht 2: Sicherungsschicht

- Aufbau und Unterhaltung einer „logischen“ Verbindung
- Zeichen- und Datenblocksynchronisation
- Erkennung von Datenblockgrenzen
- Fehlererkennung und Fehlerbehandlung
- Zugriffssteuerung auf das Medium
- sehr oft Unterteilung in 2 Teilschichten:
  - Logical Link Control
  - Medium Access Control

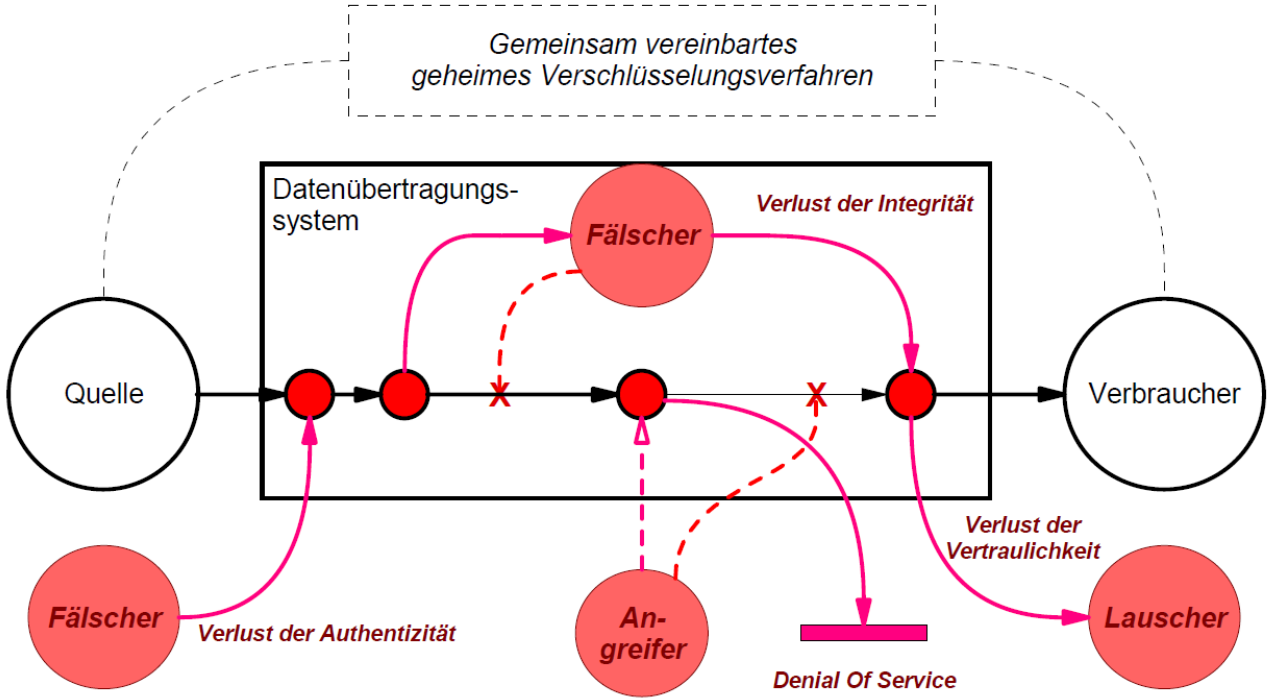
Datensicherung und Telegramme

Kommunikationsmodell nach Shannon



Datensicherung und Telegramme

Erweitertes Kommunikationsmodell



Datensicherung und Telegramme

Mögliche Störquellen durch elektrische Einflüsse

- Elektromagnetische Störfelder
  - stören das Signal, ggf. Wiederholungen erforderlich
- Potentialdifferenzen
  - Ausgleichsströme bei falscher Erdung
- Überspannungen
  - zerstören Buskoppler
- Laufzeitfehler
  - bei zu langen Leitungen
- Leitungsreflexionen
  - bei fehlenden (defektem) Abschlusswiderstand

Datensicherung und Telegramme

Mögliche Störquellen durch mechanische Einflüsse

- Bruch durch Materialermüdung
- Kurzschluss durch Materialermüdung oder durch Quetschen
- Unterbrechung, Kurzschluss an Verbindungsstellen
- Kurzschluss bzw. Unterbrechung durch Fehlbedienung
- Zerstörung durch Tiere

Datensicherung und Telegramme

Mögliche Störquellen durch thermische Einflüsse

- Schmelzen, Aufbrennen der Leitungen (Überhitzung, Feuer)
- Brechen, Reißen der Leitung durch Unterkühlung

Mögliche Störquellen durch chemische Einflüsse

- Verätzung
- Elektrische Veränderung der Leitung (Leitfähigkeit, Isolierung, ε)

Datensicherung und Telegramme

- Forderung nach ‚sicherer' Datenübertragung!
- Vermeidung von Fehlern durch
  - Hohe Signalpegel
  - Schirmung von Kabeln
  - Verwendung von Lichtwellenleitern

- Trotzdem kann es zu Fehlern kommen!
  - **Fehler müssen erkannt werden!**

Datensicherung und Telegramme

- **Entstehung eines fehlerhaften Bits**

  a) zu sendende Daten

  b) gesendetes Signal

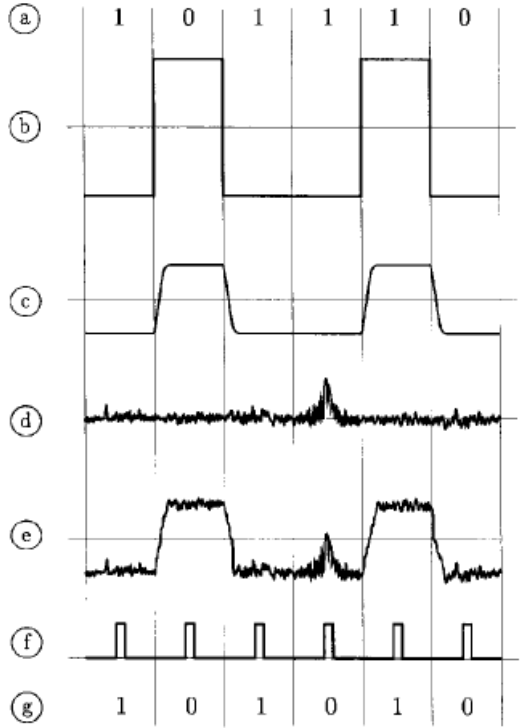
  c) gedämpftes Signal

  d) Störung

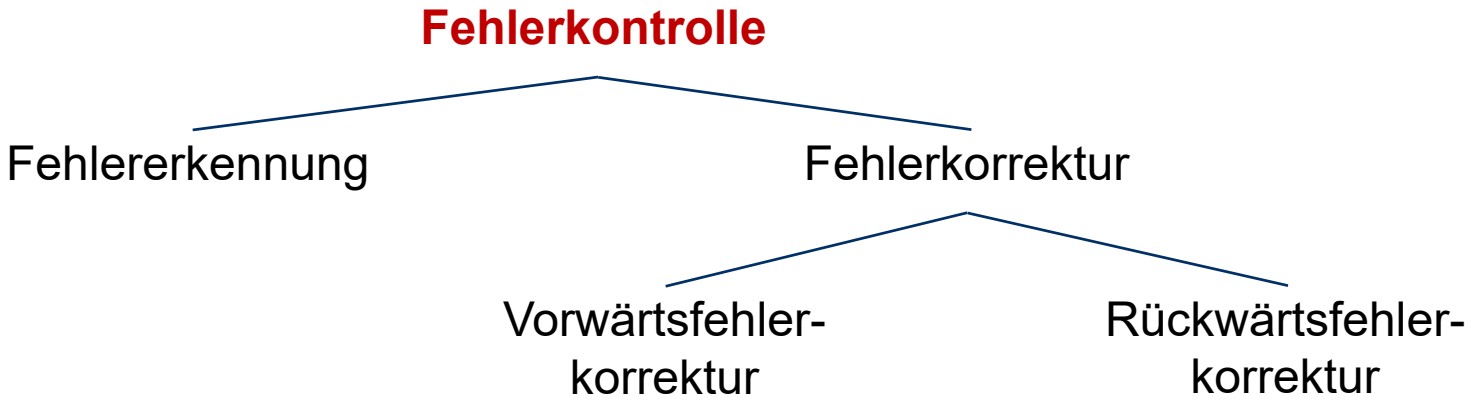
  e) gedämpftes und gestörtes Signal

  f) Signalabtastung

  g) empfangene Daten



Datensicherung und Telegramme



- Vorwärtsfehlerkorrektur (Forward Error Correction)
  - Verwendung von redundanter Kodierung, die es ermöglicht Fehler ohne zusätzliche Übertragungen zu beheben.
- Rückwärtsfehlerkorretur (Backward Error Correction)
  - Nach Erkennen eines Fehlers, wird durch weitere Kommunikation der Fehler behoben.

Datensicherung und Telegramme

Signale, Zeichen, Informationen und Bits

Einige Definitionen zur Einführung:

Zeichenvorrat (*Ziffernalfabet*)	Z = { 0, 1, 2, 3, 4, 5, 6, 7, 8 ,9}	Z = { a, b, c, d, e, f, g, h}	Z = { 0, 1}
Auftretenswahrscheinlichkeit eines Zeichens	p = 1/10	p = 1/8	p = 1/2
Informationsgehalt eines Zeichens	$H = \log_2 10$ = 3.3219 ***Bit***	$H = \log_2 8$ = 3 ***Bit***	$H = \log_2 2$ = 1 ***Bit***
Grundlage für	dezimale Codes	oktale Codes	duale Codes

Datensicherung und Telegramme

- **Kanalbewertungen**

  Ein Maß für die Störempfindlichkeit des Übertragungskanals ist die Bitfehlerrate p

$$p = \frac{\text{Anzahl der fehlerhaften Bits}}{\text{Gesamtzahl der gesendeten Bits}}$$

  Beispiel:
  von 1.000.000 gesendeten Bits sind 20 Bits gestört, das ergibt eine Bitfehlerrate

$$p = \frac{20\ Fehler}{1 * 10^6\ Bits} = 2 * 10^{-5}$$

Datensicherung und Telegramme

- Ermittlung der Bitfehlerrate ist nur experimentell möglich.
  - Typische Werte für den Einsatz im Feldbusbereich:
    - Kupferleitungen: 10^{-3} … 10^{-5}
    - LWL: 10^{-8}
- Auf Grundlage der Bitfehlerrate p und der gegeben Anzahl von Bits pro Frame N lässt sich die Wahrscheinlichkeit p_{xi} für i Bitfehler in einem Frame berechnen:

$$p_{xi} = \binom{N}{i} \cdot (1-p)^{N-i} p^{i}$$

- Hiermit kann auch die mittlere Zeit bis zum Eintreffen des Ereignisses berechnet werden, wobei v die Übertragungsrate darstellt:

$$t_{pxi} = \frac{N}{v \cdot p_{xi}} \ [s]$$

Datensicherung und Telegramme

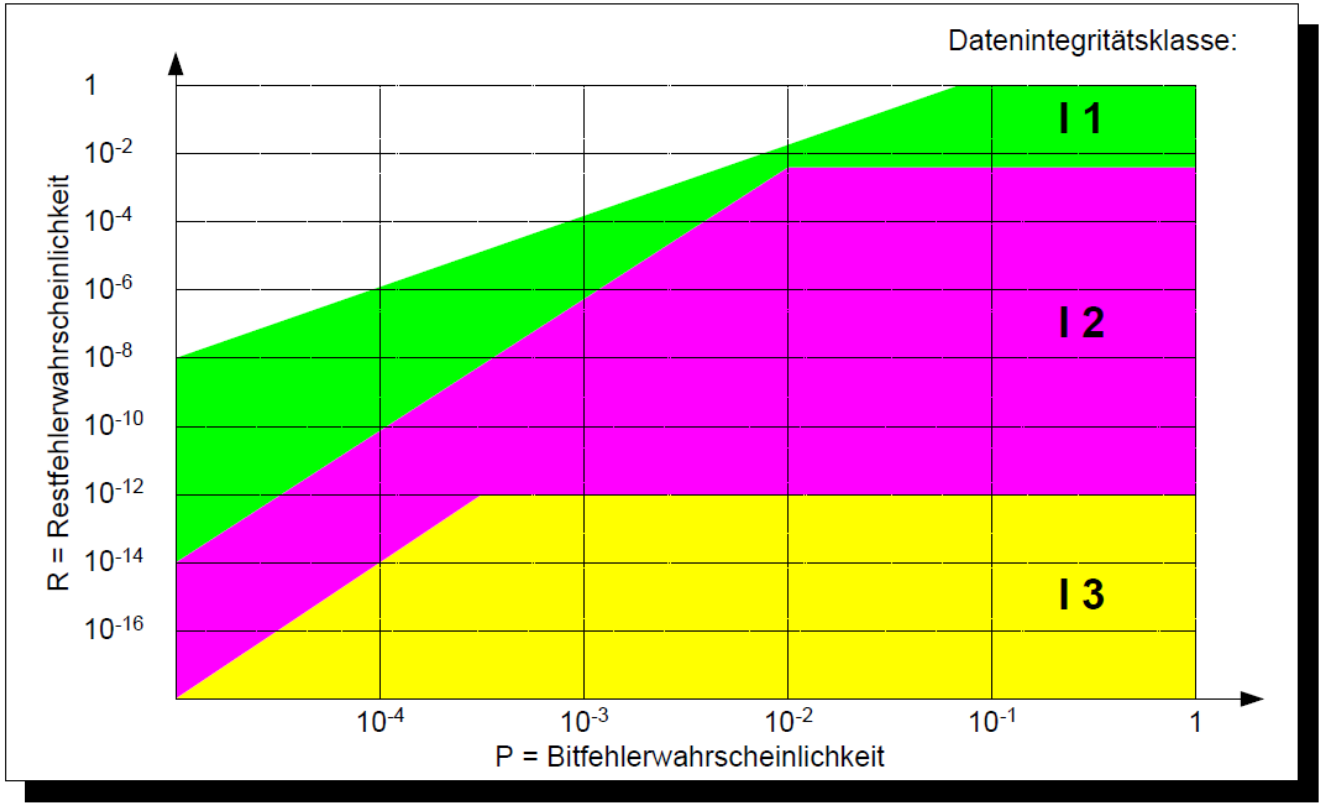
- Ein Maß für die nach Einsatz eines Fehlererkennungsverfahrens unerkannten Fehler ist die **Restfehlerrate R**:

$$R = \frac{\text{Anzahl der unerkannt fehlerhaften Bit} - \text{Kombinationen}}{\text{Gesamtzahl der gesendeten Bit} - \text{Kombinationen}}$$

- Die Restfehlerrate ist dabei auch ein Maß für die Datenintegrität bzw. Unversehrtheit der Daten.

- Die DIN 19244 schlägt hierbei 3 Integritätsklassen vor.

Datensicherung und Telegramme



Datensicherung und Telegramme

➢ Hamming-Distanz

- Maß für die Qualität eines Codes
- Vergleicht man zwei binäre Codeworte Bit für Bit miteinander, wird die Anzahl der in beiden Worten unterschiedlichen Binärstellen als Hamming-Distanz d(C) bezeichnet.

- Beispiel: 000 und 001 d(C) = 1; Unterschied in einer Stelle
  001 und 110 d(C) = 3; Unterschied in drei Stellen

Binäre Codeworte, die eine Hamming-Distanz von d(C) = 1 zueinander aufweisen, werden als benachbarte Codeworte bezeichnet.

Datensicherung und Telegramme

- **Hamming-Distanz**

Aus d(C) lassen sich direkt die Anzahl der erkennbaren und der korrigierbaren Fehler ermitteln.

Datensicherung und Telegramme

Es können bis zu f_e Fehler erkannt werden.

$$f_e = d(C) - 1$$

Es können bis zu f_k **Fehler korrigiert** werden.

$$f_k = \begin{cases} \frac{d(C)-2}{2}; & d(C) \quad gerade \\ \frac{d(C)-1}{2}; & d(C) \quad ungerade \end{cases}$$

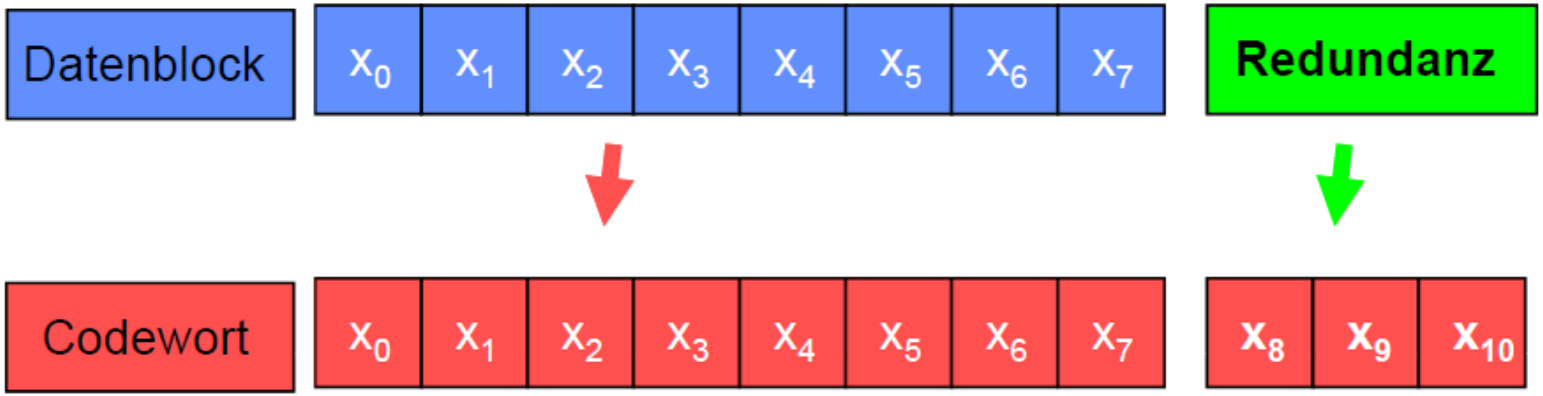
Datensicherung und Telegramme

- **Die Qualität eines Codes**

Das Prinzip des **Hamming-Codes** besteht darin, durch Verwendung mehrerer **Prüfbits** die **Fehlererkennung** so zu verfeinern, dass ein **Einzelfehler** nicht nur **erkannt**, sondern auch **lokalisiert** werden kann.

Wenn bei einem **binären Code** das **fehlerhafte Bit lokalisiert** ist, lässt es sich durch **Negation korrigieren**.

Datensicherung und Telegramme



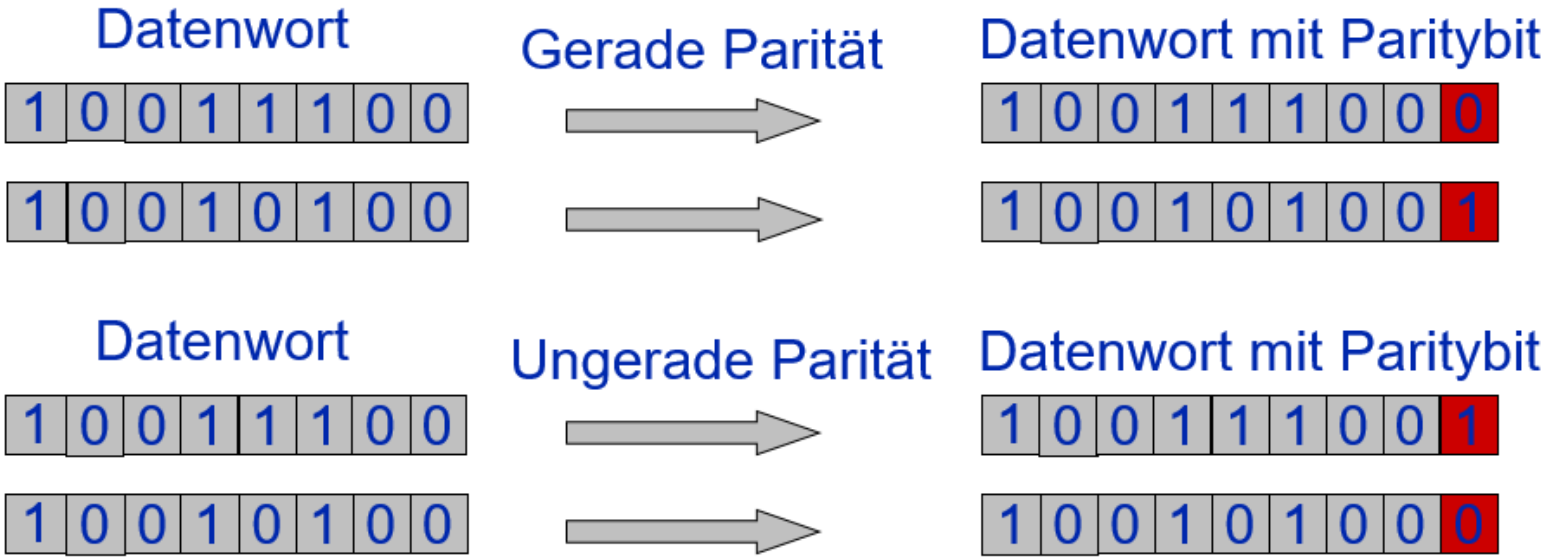
Sicherung durch Blockkodierung

- Grundprinzip der Blockcodierung: Der Datenstrom wird in Blöcke der Länge n unterteilt, nach gewissen Vorschriften wird diesen Blöcken eine Redundanz hinzugefügt, die eine Fehlererkennung ermöglicht.

Datensicherung und Telegramme

➢ Fehlererkennung mit Paritybit

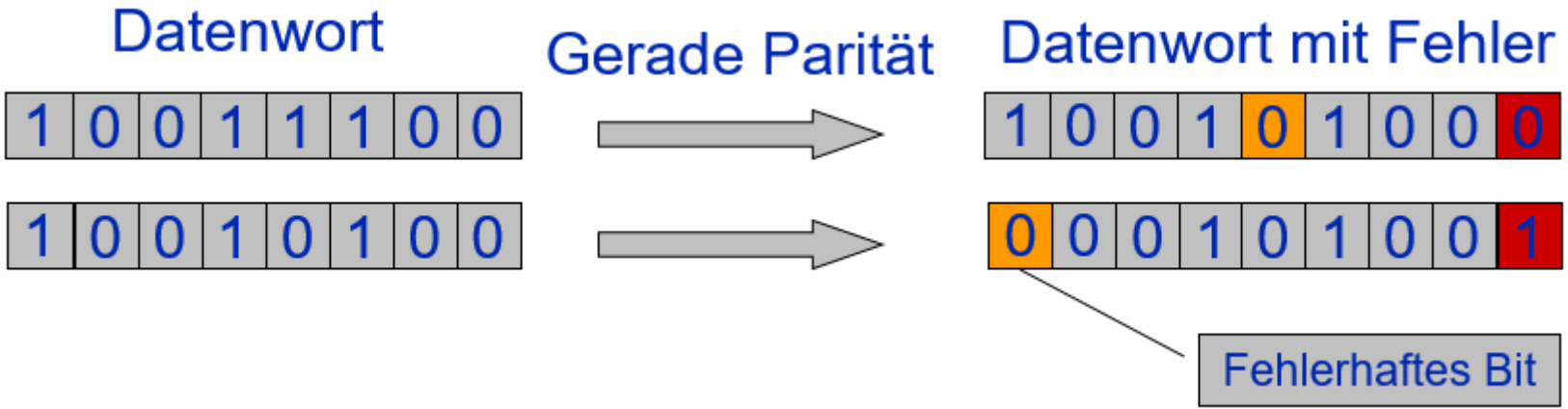
- Nach 8 Datenbits kommt ein neuntes Paritybit
- Gerade oder ungerade Parität



Datensicherung und Telegramme

- **Störung einer Datenübertragung**
  - Fehler wird erkannt. Das fehlerhafte Bit kann aber nicht identifiziert werden, also

    → Telegramm erneut anfordern



Datensicherung und Telegramme

Fehlererkennung mit Paritybit

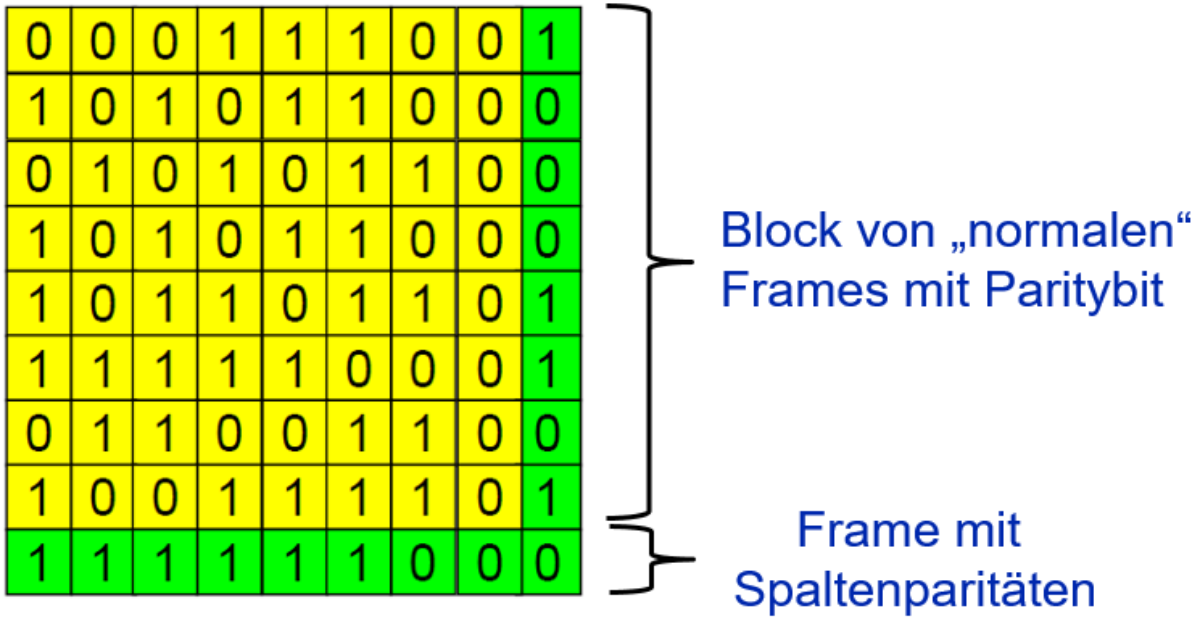
- Der Code zweier aufeinanderfolgender Zeichen unterscheidet sich in mindestens zwei Bit
  - Die Hammingdistanz ist 2
  - Je größer die Hammingdistanz, desto größer die Chance Fehler zu entdecken:

$$f_e = d(C) - 1$$

  - $f_e = 2-1 = 1$
  - Bitfolgen mit 3, 5, 7, etc. Fehlern lassen sich auch als fehlerhaft erkennen. Eine Aussage darüber, wie viele fehlerhafte Bits vorliegen kann nicht getroffen werden.

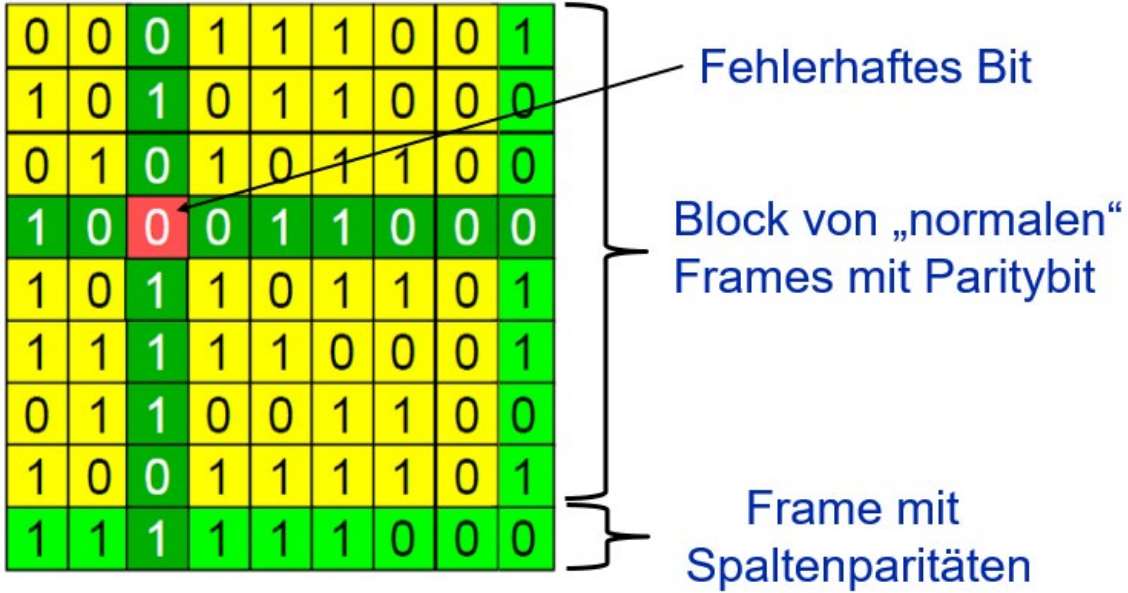
Datensicherung und Telegramme

- **Blocksicherung mit gerader Parität**



Datensicherung und Telegramme

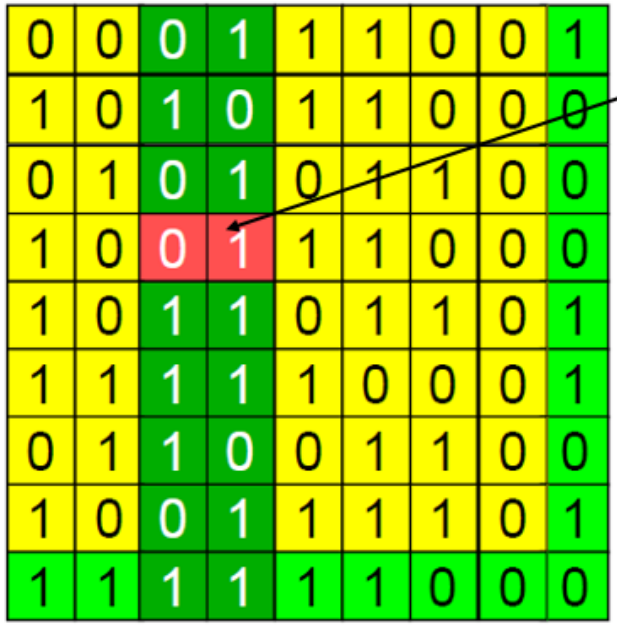
- **Blocksicherung mit gerader Parität**



Die Anzahl der ‚1‘ in Zeile und Spalte stimmt nicht mit dem Paritätsbit überein, der Fehler wird erkannt und kann korrigiert werden.

Datensicherung und Telegramme

- **Blocksicherung mit gerader Parität**

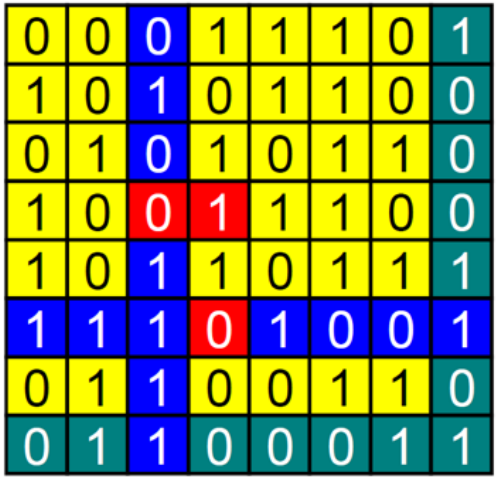


2 Bit gestört,
Paritätsbit richtig

Die Anzahl der ‚1' in Spalte 2 und 3 stimmen nicht mit dem Paritätsbit überein, der Fehler wird erkannt, kann aber nicht korrigiert werden, weil die Zeile nicht lokalisiert werden kann.

Datensicherung und Telegramme

- **Blocksicherung mit gerader Parität**



Achtung!
Kann zur Korrektur „falscher Bitstellen“ führen

Datensicherung und Telegramme

Fehlererkennung mit Prüfsumme

- Daten werden um eine Prüfsumme erweitert.
- Mehrere Zeilen/Daten werden zusammengefasst und zusammen gesichert.
- Die Zeilen werden addiert, ein Übertrag ignoriert.
- Beim Empfänger wird geprüft, ob die übertragene Summe mit der berechneten Summe übereinstimmt.
- Meist in Kombination mit Paritybit angewandt

Datensicherung und Telegramme

- **Fehlererkennung mit Prüfsumme**
  - **Beispiel**

Gesendete Daten	Datenbits	Paritybit
Datenbyte 1	0011 1001	0
Datenbyte 2	1111 0100	1
Prüfsumme	0010 1101	0

Um 16 bit sicher übertragen zu können müssen

2*(8+1)+1*(8+1) Bit = 27 Bit

übertragen werden!

Hammingdistanz ist 4 → 3 Fehler können sicher detektiert werden!

Datensicherung und Telegramme

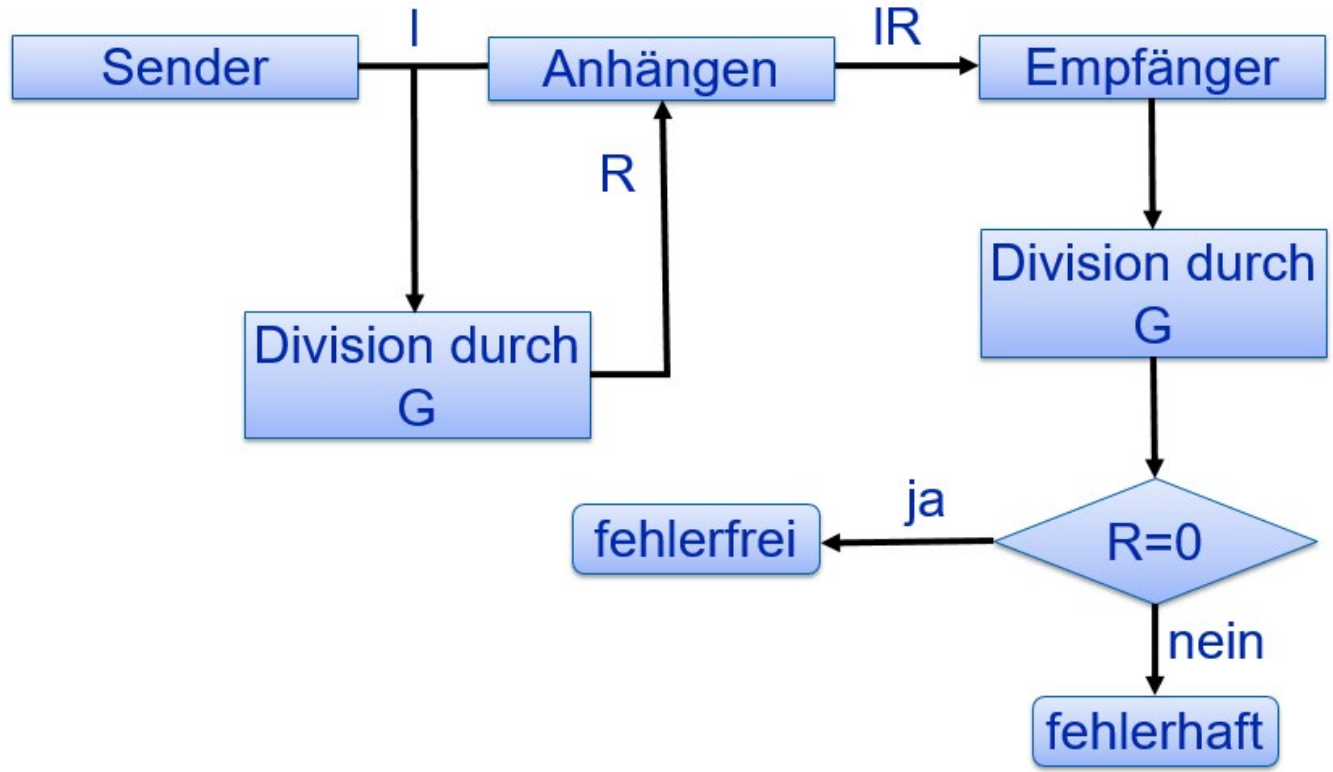
Übersicht Cyclic-Redundancy-Check/CRC:

- Die **Information I** wird als **Zahl** von beliebiger Länge **interpretiert** und durch ein **Generatorpolynom G dividiert**. Der resultierende **Rest R** wird an die Information I **angehängt**.
- Auf Seiten des **Empfängers** wird der **Codevektor IR** durch dasselbe **Polynom G dividiert**. Bei einer **fehlerfreien Übertragung** ergibt die **Division den Rest 0**.
- Die zu übertragenden **Codeworte** haben eine **Hamming-Distanz** von **>= 4**.
- **Relativ komplizierter Algorithmus**, aber **einfach** in Hardware oder Software **zu realisieren**.

- Interbus, Profibus-PA, LON: 16-bit-FCS
- CAN: 15-bit-FCS

Datensicherung und Telegramme

Cyclic-Redundancy-Check/CRC:



Datensicherung und Telegramme

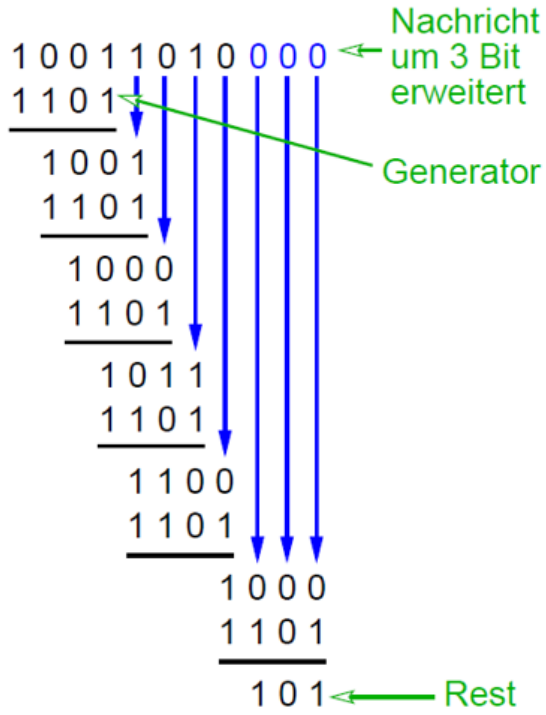
Cyclic-Redundancy-Check/CRC:

- **Idee:**
  - ein b Bit langer Bitstrom L wird als Polynom L(x) des Grades b-1 aufgefasst, z.B. werde 1001 1010 interpretiert als $x^7+x^4+x^3+x$
  - Es wird ein Generatorpolynom G(x) vom Grad k gewählt.
    - z.B.: G(x) = 1101 → x^3+x^2+1
  - Die Nachricht L(x) wird um k Prüfbits erweitert:
    - 1 0 0 1 1 0 1 0 0 0 0
  - Dann: Polynomdivision der erweiterten Nachricht durch G(x) mit Modulo-2-Arithmetik

Datensicherung und Telegramme

Cyclic-Redundancy-Check/CRC:

- Beispiel:



- Versendete Nachricht P:
- 10011010 101
- Die Nachricht ist durch den Generator ohne Rest teilbar !

Datensicherung und Telegramme

Cyclic-Redundancy-Check/CRC:

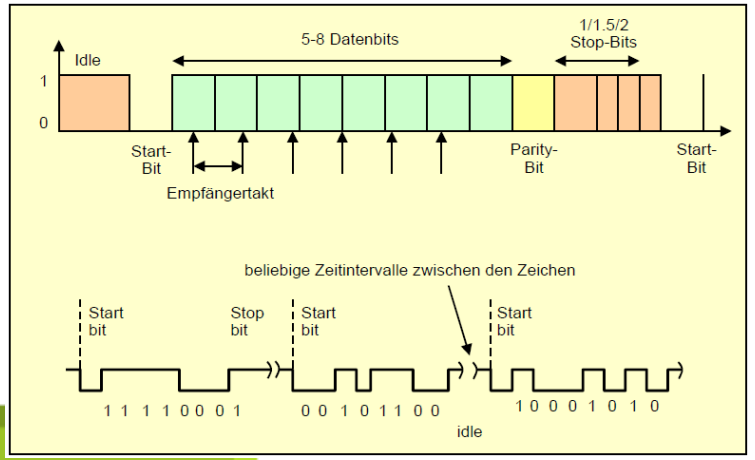
- **Wahl des Generatorpolynoms?**

  - So, dass möglichst viele Fehler erkannt werden!
    - Beispiel für ein übliches CRC-Polynom:
      - CRC-16: $x^{16} + x^{15} + x^2 + 1$

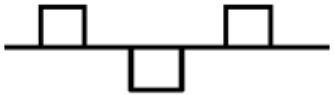
- CRC-16 erkennt:
  - alle Ein-und Zweibitfehler
  - alle Fehler mit ungerader Bitanzahl
  - alle Fehlerbündel mit Länge $\leq$ 16 Bit

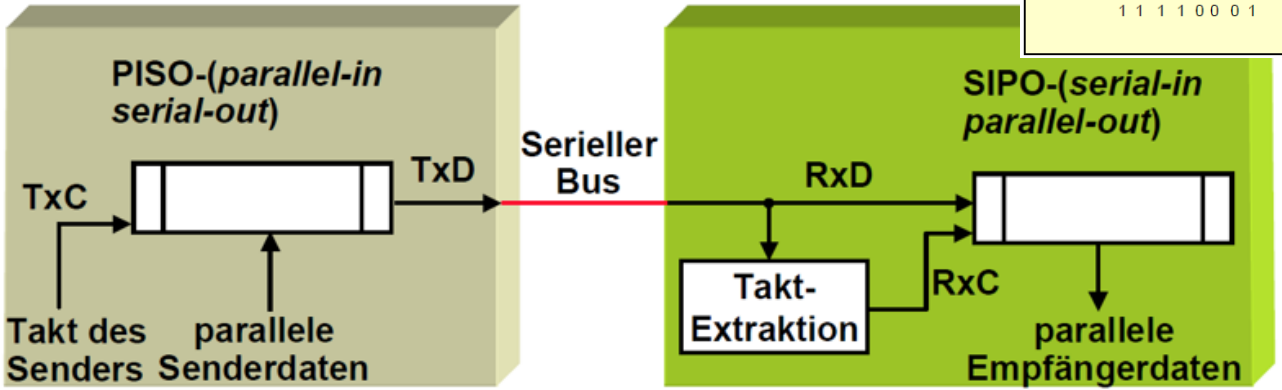
Ausblick

Synchronisation & Framing



- **Beispiel: Codierung von 1 0 1 →**





Hochschule Düsseldorf
University of Applied Sciences
HSD
ZDD - DAISY

UbiComp – Teil 5: Netzwerktechnik und industrielle Kommunikation II

Fragen?

Prof. Dr.-Ing. Dorothea Schwung